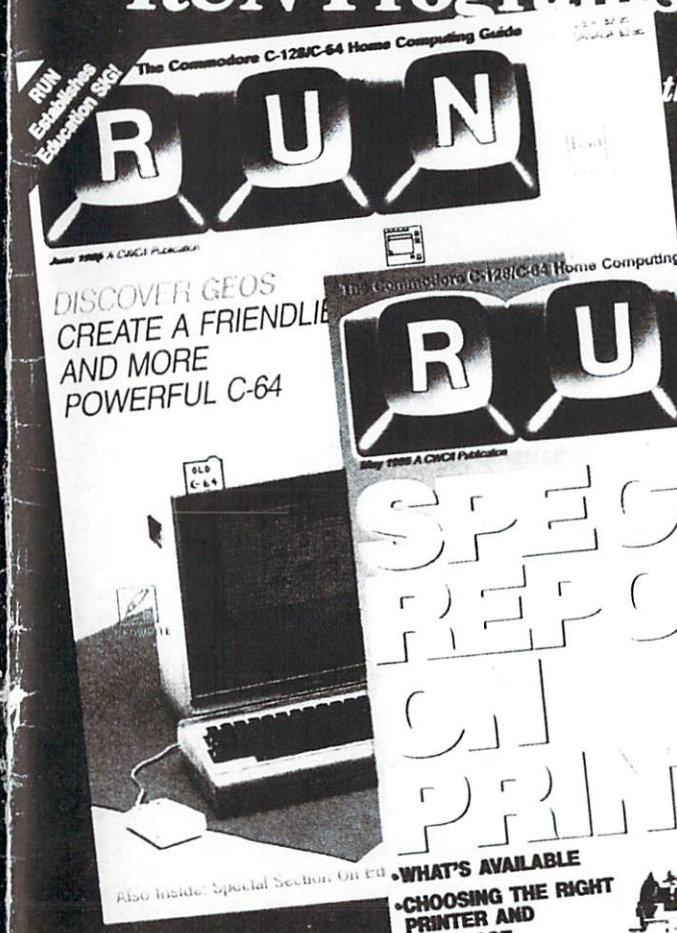
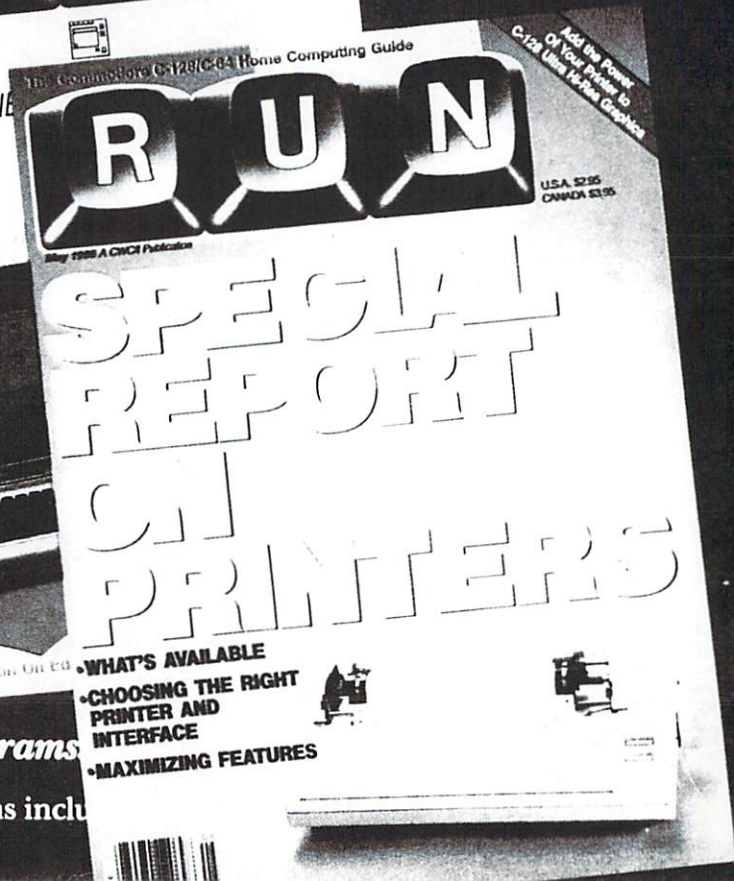
May/June 1986 Edition

RUN

RUN Programs on Disk

*the C-64 and C-128**





Plus: Bonus Programs

*128 mode programs inclu

Introduction

May–June '86 ReRUN

This ReRUN contains all of *RUN* magazine's programs from May and June 1986, as well as two bonus programs. (In the brief descriptions below, article titles are in parentheses if different from the filename listed in the disk directory.)

The menu programs on this disk will let you load and run your C-64 and C-128 programs with a single key press. Menu programs are meant for running stand-alone programs, not sequential files or sub-modules of a main program. This is why you should carefully read the directory page and the article for each program *before* running it.

For example, C-128 Ultra Hi-Res Graphics is an update of a previously published program (see Ultra Hi-Res Graphics, in *RUN*'s February issue or January-February ReRUN disk), and it has several modules that cannot be run without using the original program. The three programs on this disk, UH.DATA 128, MAKEUH V1.1, and UH.PIC CONVERT, are enhancement modules that add several new commands and a print option. You have to carefully follow the instructions for installing these three modules, so read the article first.

We have several programming utilities this time. DISK READER lets you read just about any file on your disk, whether it be sequential, Basic, or machine language. It will also give you a printout of machine language programs.

With COMMA GENERATOR (Easy Data Entry), you can type data statements more easily. This C-64 utility automatically inserts commas and the word Data into your programming lines.

QUICK MERGE is particularly helpful for C-64 programmers who have collected a library of subroutines. Instead of retyping the same code, this utility automatically loads and merges the subroutine code into your programs.

TOKENIZER (Easy Disk-File Conversion) will be appreciated by those of you who download programs in sequential file form and need to convert them back into usable Basic form. The article also includes a quick way to convert Basic programs into sequential files.

80-COL CHARACTER (Give

Your C-128 More Character) is a handy 128 mode program that lets you create your own character fonts for use in other 128 mode programs. You could, for example, use this to create customized fonts for use with Ultra Hi-Res.

We also have a game for you this time. SWISH is a C-64 computerized basketball game, in which you and an opponent play one-on-one.

Education is covered in this edition with ARITHME-SKETCH, a math and drawing program for younger students. This math program provides positive reinforcement for correct answers by awarding time to be used in the drawing module of the program.

LABEL MAKER is a C-64 application that lets you create and print out custom-designed mailing labels.

Lastly, we have two bonus programs. C-64 owners have a collector's database program, called FIND IT. For C-128 owners, there's a whodunit mystery, called MURDER MYSTERY (Murder by Byte).

See you next time with programs from *RUN*'s July and August issues.

Margaret Morabito
Technical Manager
RUN *magazine*

Directory

* You must have Ultra Hi-Res, Part 1 to use these three programs. See January–February 1986 ReRUN or February 1986 issue of *RUN* magazine.

£ Bonus program!

How To Load

This ReRUN provides menu programs for both C-64 and C-128 users. C-64 users should type LOAD "MENU 64" and RUN. C-128 users need only press the shift and run/stop keys. These menus will display all of the programs on the disk and let you run them with a single key press.

If you don't want to use the menu programs, then follow these guidelines for loading individual programs.

C-64:

To load a C-64 program written in Basic, type:

LOAD "PROGRAM NAME",8

and then press the return key. The drive will whir while the screen prints LOADING, and then READY, with a flashing cursor beneath. Type RUN and press the return key. The program will then begin.

Machine language (ML) programs will be loaded with LOAD "PROGRAM NAME",8,1. When these occur, they will be clearly marked for you on the directory page.

C-128 in C-64 mode:

All C-64 programs can be run on the C-128 as long as your computer is in C-64 mode. This is accomplished by typing GO64 after powering up your C-128. Answer Y and press the return key in response to the "Are you sure?" prompt. When in C-64 mode, use the instructions above for loading C-64 programs.

C-128 in C-128 mode:

All C-128 mode programs are clearly labeled on the directory page. Your C-128 must be in C-128 mode to run these programs. All programs will work in both 40 and 80 columns, unless they are specifically marked otherwise.

To load a C-128 mode program, just press f2 and type the name of the program that you wish to load. Use the exact name that appears on the directory page. When the program has loaded, a READY prompt will appear. Beneath this is the flashing cursor where you should type RUN and press the return key. The program will then run.

NOTES:

Before loading any program, always refer to the article in this booklet for special instructions.

It is a good idea to make a copy of your ReRUN disk before running the programs. You can accomplish this by using the disk backup programs that Commodore provides for the 1541 and 1571 disk drives.

ReRUN Staff

Technical Manager/Editor: *Margaret Morabito*
Technical Editor: *Tim Walsh*
Art Director: *Glenn A. Suokko*
Design and Layout: *Karla M. Whitney*

Swish!

By Mark Jordan

RUN It Right

C-64; C-128 (in C-64 mode); two joysticks

If the title of this program conjures up images of braying, sweating, racetrack animals, then it's obvious you're not from Indiana. Everybody here knows that Horse is a backyard basketball game. And it's fun because only one (and everybody's favorite) basketball skill is required—shooting.

The rules are simple: Each time you make a basket, your opponent must duplicate the shot. If he misses, he gets a letter from the word *horse*. If he succeeds, neither one of you receives a letter, and you simply continue making your shots. Once you miss, though, your opponent gets a free shot and a chance to turn the tables. The first player to spell out *horse* loses.

The computerized version of this game follows the same format as the backyard game. It requires two joysticks, two players and at least one good shooting eye. You control player 1 with a joystick plugged into port 2; your opponent controls player 2 with a joystick plugged into port 1.

You begin the game by moving your player anywhere on the court to shoot. You press the fire-button to position the ball for the shot, then press the button again, this time holding it down to control the ball's arc. When you feel the ball has achieved just the right height to begin its descent, you release the button.

The ball completes its upward climb with a neat loop and begins to fall. It nears the basket and, *swish*, falls through. A bell rings, and the word *good* appears beneath the scoreboard. The ball continues falling until it hits the floor, where it begins bouncing.

Now your opponent, player 2, hustles over to retrieve the ball and starts dribbling like mad. He quickly moves to the spot from which you made your shot. He must position himself within 16

pixels of your x,y shooting co-ordinates, or the shot won't go, and he will get a letter.

So, player 2 places himself where he thinks you were just positioned (you were smart; you moved away from your shooting location), and presses his fire-button. His sprite switches from dribbling to the ready-to-shoot position. He again presses and holds the fire-button, and the ball arcs gracefully to the basket. He releases. The ball descends.

Boing! It strikes the back of the rim and bounces high into the air. Player 2 sucks in his breath and watches as the ball comes down—*and through!*—the hoop. No H this time.

The game continues as player 1 grabs the ball and moves to a new location to shoot again. As long as he keeps making his "free" shots, he cannot lose. Once he misses, though, player 2 gets the free shot, and the tables are turned.

DO IT WITH STYLE

Those are the basics; here are the embellishments. First, in computerized Horse, you'll find that you can change your shooting style. When you press the button the first time, you'll not only discover that you've maneuvered your man into the shooting posture, but that moving the joystick will no longer affect his screen position; he is frozen to that spot until the shot is completed.

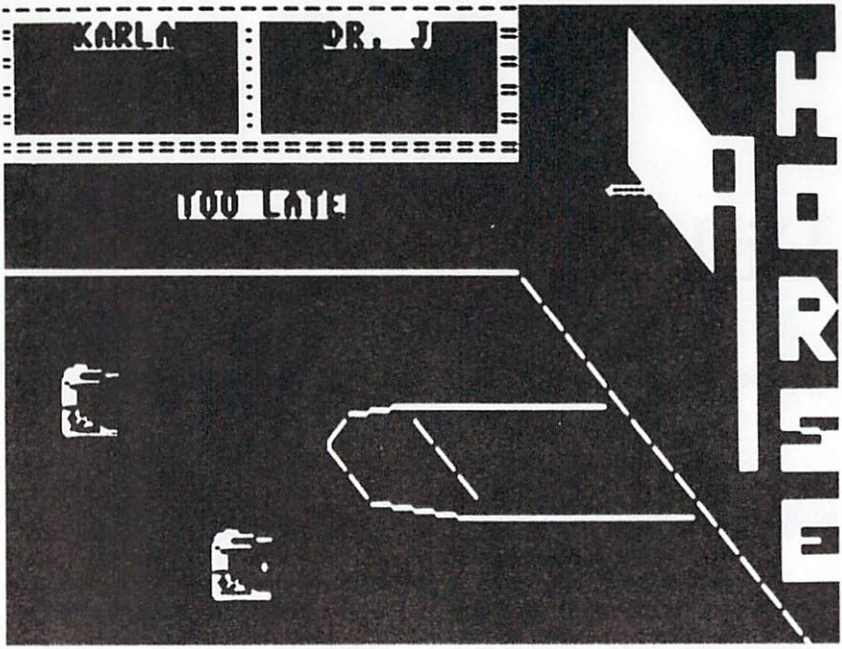
Now, by pushing upward on the stick while pressing the button to start the shot, the ball will rise on a much steeper path than before. This high-arc shot is especially good for close-in shots, such as lay-ups. Note that you can release the stick after you have pressed the button. The high arc is selected at the instant you press the button and will remain that way throughout the shot.

If you'd like a low-arc shot, just point the joystick downward while pressing the fire-button for the second time.

The advantage of choosing a high- or low-arc shot is that, if you make the shot, your opponent will be forced to shoot the same arc. In fact, it won't matter if he pushes the joystick or not—the arc will mimic yours automatically.

The high-arc shot can be tricky because it sometimes arcs above the viewing range of the screen. You have to depend on your sense of timing to choose the exact moment to release the button. The low-arc shot is difficult because, just as in the real game of basketball, its sharp angle requires a finer shooting touch. Not only that, but low-arc shots are very difficult to bank in.

Did I mention bank shots? Yes, you can bank the ball off the backboard. If you choose the bankshot option at the opening



of the game, all bankshots made must be duplicated with a bankshot, or the non-banking shooter will get another letter, whether or not he makes his shot. However, until you become familiar with Horse, I recommend you press N (for no) whenever the bankshot option is displayed.

Besides bankshots, you are given two other options at the game's beginning: You can adjust the ball's flight speed, and you can change the level of difficulty in making the shot. Both of these options let you control the difficulty of play.

Finally, this version of Horse features something the backyard game doesn't have—a shot clock. When the ball hits the floor from the previous shot, you have exactly seven seconds to get your shot off. The clock is visible. The reason for having this feature is to put a little hustle into the game. It forces you to grab the rebound and move quickly into your opponent's old position. You can change the length of time by changing the value of CT in line 260 of the program.

Well, that's it. I hope you have fun horsing around. R

Disk Reader

By Paolo Agostini

RUN It Right

C-64; C-128 (in C-64 mode)
Disk drive; printer optional

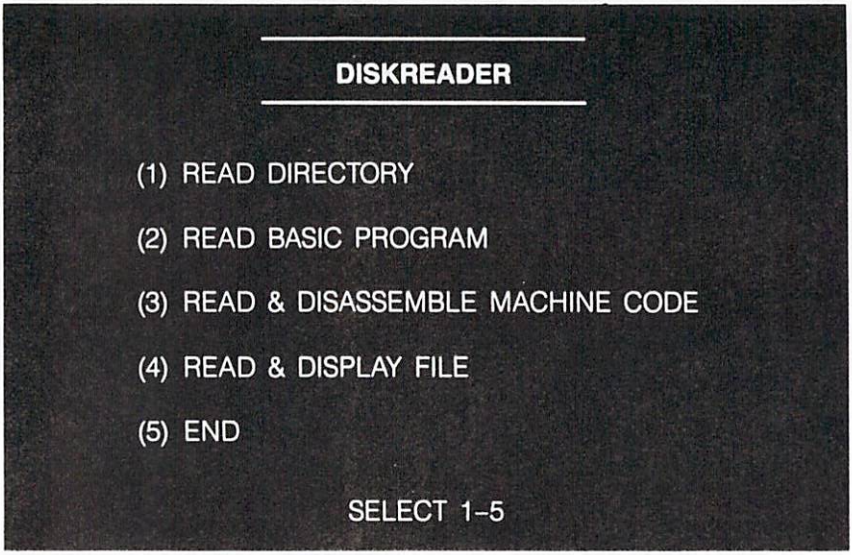
Some Basic and machine language programs are extremely difficult to examine or decipher. There are various reasons for this. Some programs are loaded in memory sections where they overlap the machine code monitor; some are "hidden" in the RAM under ROM; some are copy-protected to prevent us from examining them.

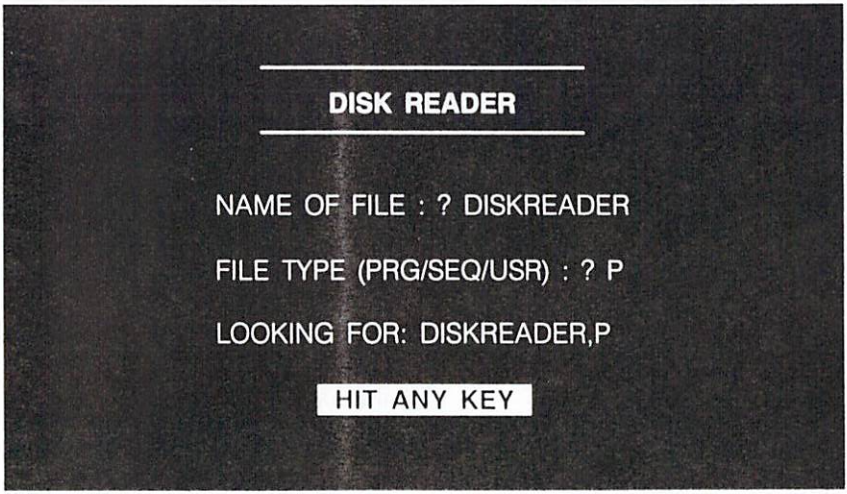
Now, with Disk Reader, you can read data from almost any disk (the exceptions being those that are copy-protected). You may have the output printed to both the screen and the printer. Disk Reader has the following options.

1. The ability to read the disk directory without loading it into the computer's memory.
2. The ability to read any Basic program from disk, thus reconverting tokens into keywords.
3. The ability to read any machine code program, printing it in the form of addresses, hex numbers and assembly language opcodes—just as you'd see it with any ordinary machine language monitor.
4. The ability to read any sequential file from disk, skipping unprintable characters.

You may stop the flow of data at any time by pressing the S key, which stops the output and returns you to the main menu. Or, you may use the shift or Commodore key, which pauses the output for as long as it's pressed. (For especially long pauses, use the shift-lock key.)

To end the program, use option 5 of the main menu; should you use the run/stop key, the communication channels between the computer and disk drive (and eventually printer) will not be properly closed. R


DISKREADER
(1) READ DIRECTORY
(2) READ BASIC PROGRAM
(3) READ & DISASSEMBLE MACHINE CODE
(4) READ & DISPLAY FILE
(5) END
SELECT 1-5


DISK READER
NAME OF FILE : ? DISKREADER
FILE TYPE (PRG/SEQ/USR) : ? P
LOOKING FOR: DISKREADER,P
HIT ANY KEY

Label Maker

By Mike Konshak

RUN It Right

C-64; C-128 (in C-64 mode)
Disk drive; 1525 printer

With the Label Maker program, you can create a label, then generate as many copies of it as you desire. Printing uniform labels gives your disk collection, personal possessions, and so on, a more professional appearance. And, if you use name tags at club functions, it's nice for them to be identical. You could also serialize your disk labels to keep track of program revisions and the like.

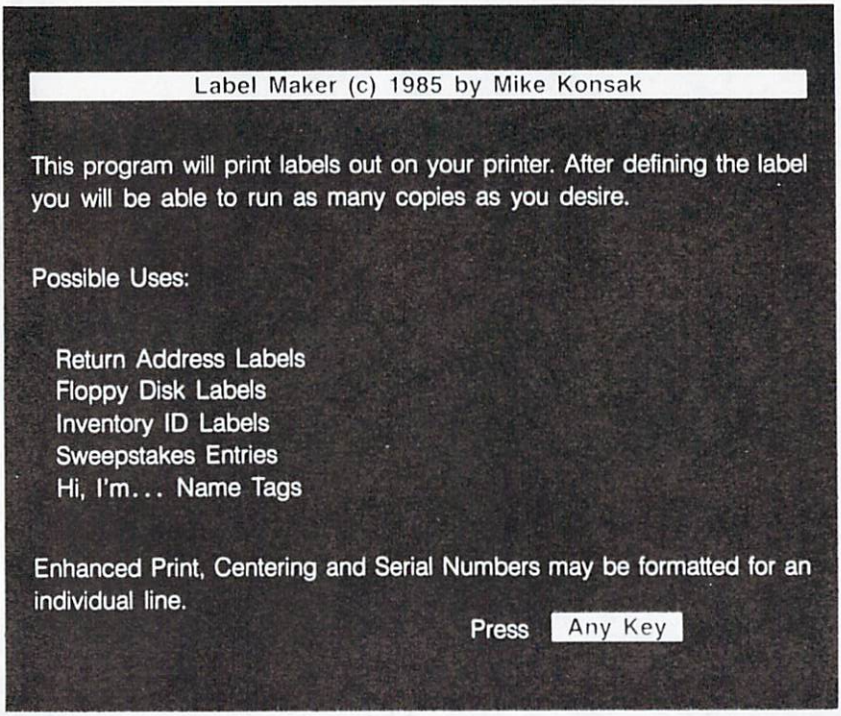
Label Maker was designed to print on one-up labels. If you have two-up labels, you can simply flip the label sheet over. If you ever need an enormous label, the program is capable of designing labels that use up to 66 rows and that are 70 characters wide.

That is the same size as a full piece of paper, but I would not recommend this as a one-page word processor; it would be too difficult to edit the lines.

Labels are generally separated by one line, or vertical space. Standard-size labels, which are most commonly available, are $^{15}/_{16}$ of an inch wide by three to five inches long, and can hold five printed lines. Labels that are 3½ inches long are the most universal, because they can print 34 characters on one line, which is the normal number for the length of an address.

When designing your labels, you'll be asked to enter the number that's one greater than the maximum number of rows your label can contain. For example, if your label can print five lines, then enter 6. Next, you'll be asked for the number of rows you actually want printed. If your label can print a maximum of five rows, you should enter a number from 1 to 5; if 8, then 1 to 8.

Lastly, you'll be asked for the possible number of characters per row. This normally defaults to 34, for labels that are 3½ inches long. If you have labels of a different length, or if you've put your printer into Compressed mode or pitch, then you should

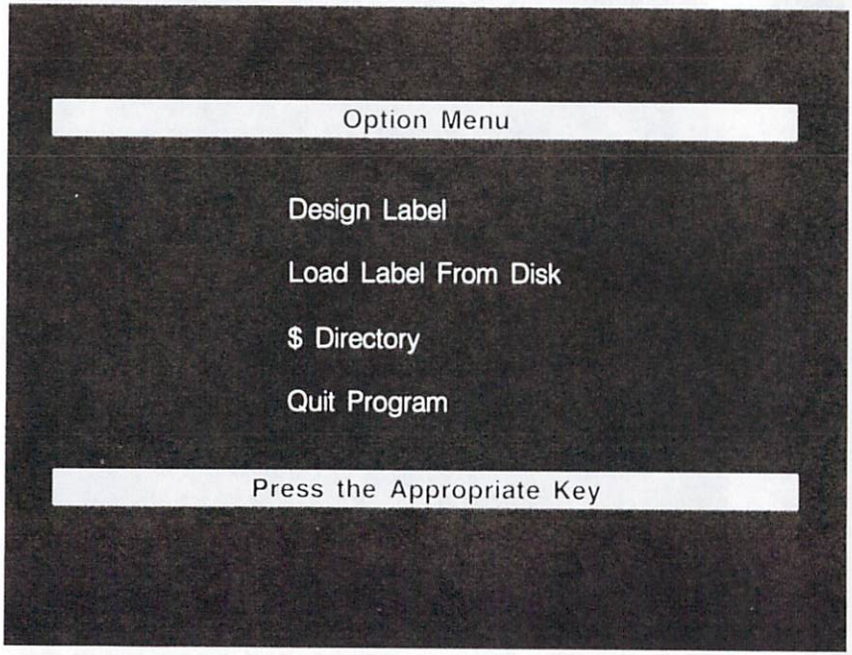


adjust the desired number of characters per row accordingly.

You may save your labels on disk for recalling at a later date. Each label design is saved in individual sequential files. The filenames are preceded by a special code, LM] , which lessens the confusion if there are other sequential files on the same disk. Filenames cannot exceed 12 characters, since the code takes up four characters.

Besides printing text in standard pitch and in upper- and lowercase letters, Label Maker provides three special modes. Only one of these capabilities may be used in each line or row.

1. *Enhanced or Double Wide Print.* Most printers, including those by Commodore, have this capability. A line, or row, on the label will be printed in enhanced print if the data for the row is preceded by an up-arrow symbol (↑)—the exponent symbol to the left of your restore key. Since you are printing characters that are twice as wide, you can only



use one-half the number of characters per line. A data line would look like this:

1 ? †Club Disk Library

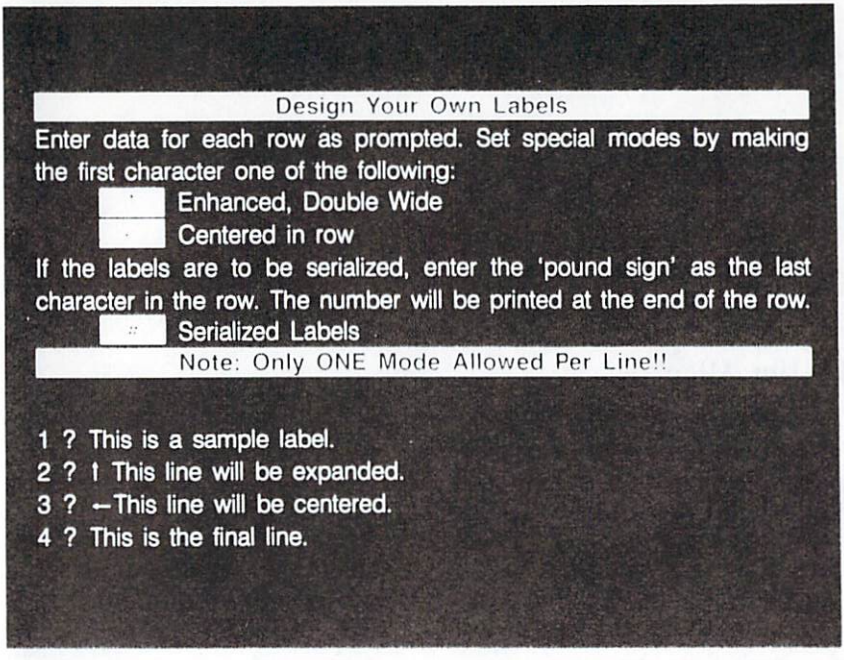
2. *Centering of Text.* Text will be centered on the label if the data is preceded by a left-arrow symbol (←)—the key in the upper left-hand corner of your keyboard, just above the control key. To activate centering, a data line would look like this:

2 ? >Property of the Computer Club

3. *Serializing, or Auto-Numbering.* A serial number may be printed at the end of a line by adding the number sign (#) at the end of the data for that row. Numbering must start at any integer greater than zero. A data line that will print a serial number looks like the following:

3 ? 12/85 Library I.D. Number#

Since the Basic Input statement is used for data entry, you may not use commas, colons, semicolons or quotation marks in your lines of data. If you exceed 70 characters, you may experience input problems, because the C-64 screen editor only accepts two lines when using input.



If you wish to replace a line of data with a blank line, then you must enter some single character (such as >) as the first character, then use the space bar to erase the remaining characters. Once you have saved and reloaded the label, the character you initially used for this deletion process will not appear when printing.

Once you've designed a label (be sure to save it) and have chosen to start printing it, you'll see the label on the screen. Double-width lines will be shown reversed; the text will be centered as it will appear; and if you've chosen serializing, a dummy serial number will be displayed to remind you that you've done so.

If you decide that that is not the label you had intended to print, enter a 0, or just press the return key without any entry, to exit the routine. The prompts in Label Maker are self-explanatory, so you shouldn't experience any problems.

If you run this program on the C-128, you must put it in C-64 mode or set up a 40-column window. Otherwise, the Data Input routine will not work correctly. R

Easy Data Entry

By Jim Allen

RUN It Right

C-64; C-128 (in C-64 mode)

While typing in line after line of numeric Data statements, your fingers spend most of their time on the top row of keys, except when typing in the word DATA and commas. This machine language program, Comma Generator, changes the space bar into a comma generator, making it easy to type in commas, and uses the back-arrow key (←) to type in the word DATA.

Once you start typing in Data statements, you won't have to move your fingers off the top row of keys until you are through. This program works only for Data statements that don't contain spaces.

USING THE PROGRAM

When you enter a program with a great number of Data statements, load and run the Comma Generator program first. This way, when you're ready to enter Data statements, you won't have to save and reload the program on which you're working.

When you are ready to enter numeric or alphanumeric data (without any spaces), type: SYS679 and hit the return key. Thereafter, when you press the space bar, a comma will appear on the screen, and when you press the back-arrow key (←), the word DATA will print on the screen.

To disable the utility, hit the run/stop and the restore keys simultaneously. You can reenable the program with SYS679 without rerunning the Basic loader. R

Arithme-Sketch

By James Pellechi and Ted Jean

RUN It Right

C64; C128 (in C64 mode)

Arithme-Sketch combines arithmetic and drawing in a unique way. The menu prompts you to select addition or subtraction and then a grade level from 1–5. The first-grade level selects problems with one-digit numbers, the second-grade level two-digit numbers, and so on.

The program displays a sheet of wide-lined paper on a clipboard. Each problem appears on the paper in large block numbers. Sound and visual effects accompany answers. After five problems have been answered, the program moves to the drawing screen.

Drawing is done with a joystick plugged into port 2. The amount of time allotted for drawing depends on how many problems were answered correctly; each correct answer earns one minute of drawing time. The time remaining is displayed at the top of the screen, and when it runs out, your artwork is saved in memory, and the clipboard displaying the main menu appears for another round of math.

While in Drawing mode, you may use the following keys or combinations thereof.

Joystick fire-button—Pressing the button turns a square on. Releasing the button allows you to move the cursor anywhere on the screen.

Numbers 1–8—Selects one of eight drawing colors.

Shift—Erases whatever square the cursor is currently on.

F1 key—Saves the screen to memory; drawing time then continues.

F2 key—Replaces current screen with screen last saved; drawing time then continues.

CTRL-f1—Erases current screen (saved screen remains unchanged); drawing time then continues.

Shift Lock—Automatically erases individual squares on the screen as the cursor is moved by the joystick.

CTRL-logo—Aborts Drawing mode; saves final screen to memory; returns to menu. R

Easy Disk-File Conversion

By Ken Stange

RUN It Right

C-64; C-128 (in C-64 mode)

The average programmer uses two kinds of disk files: program and sequential. The essential difference between them is that Basic reserved words and commands, carriage returns, etc., are tokenized (abbreviated into one-byte code) in program files, whereas sequential files read just as ordinary text.

When you type a command such as Print and press the return key, the command is tokenized. It is the carriage return that initiates the tokenization. If you write a one-line program and, instead of pressing the return key, you cursor down, type RUN and hit the return key, nothing happens. This is because the leading line number and a carriage return are needed to enter a program line and tokenize the Basic commands in that line.

Because of tokenization, program files are stored in more compact form than sequential files. You cannot directly load a program file into a word processor or terminal buffer. If you could, you'd see an almost incomprehensible mixture of text and the odd characters that are the tokens, the codes, for the Basic commands.

NEED FOR SEQUENTIAL FILES

You learn about program files as soon as you start computing, in connection with loading and saving programs, and they are, superficially, the easiest to manipulate. In some senses, however, sequential files are actually easier to use. The storage format is simpler, the manipulation of the contents easier and transportability greater.

To take advantage of these characteristics, there are times when it's helpful to translate program files into sequential files and then back again. Let's say

you want to send a program to a friend via modem. Unless you have a terminal program (such as XMODEM, for example) that supports a standard protocol, sending the program as a sequential file, character by character, is sometimes the only way to accomplish it.

This involves de-tokenizing your program, and, once you've sent it as a sequential file, your friend will have to re-tokenize it. In other words, you need a way to translate, or convert, a program file into a sequential file and then back again.

Another need for this conversion arises in writing programs. If your word processor can read sequential files, polishing a program on the word processor is infinitely easier than doing so in the Edit mode. (You can't, of course, actively debug it. But you can easily renumber lines and otherwise fine-tune its structure.)

Also, unless you like to test every few lines of code as you write them, creating the first version of a program is easier with a word processor.

In working out a way to set up this conversion, I first investigated different methods of storing a program as a sequential file.

The easiest way involves using the technique of listing to a device other than the screen or printer, as a listing is simply a non-tokenized printout of a program. You can list to the disk drive, and the procedure is virtually the same as listing to a printer.

It's a bit unnerving when you do this for the first time, because the disk drive doesn't stop whirring after the file is written. The only cue that the listing is complete is the return of the blinking cursor to the screen.

PROGRAM FILE TO SEQUENTIAL FILE

Here are the instructions on converting a program file to a sequential file. Note: Don't try to compress all of these commands into one line.

1. Load and list your program; then insert the disk on which you wish to write the program as a sequential file.
2. Type: OPEN15,8,15,"I" <return>
3. Type: OPEN8,8,8,"0:<filename>,S,W" <return> (Be careful not to use the same name as *any* kind of file on that disk.)
4. Type: CMD8:LIST <return>
5. When the cursor reappears, type: PRINT#8:CLOSE8:CLOSE15 <return>

If you check your disk directory at this point, you'll see that your program is now in sequential form, and capable of being manipulated inside a word processor or sent to a friend via a modem.

SEQUENTIAL FILE TO PROGRAM FILE

The next step is the translation of the sequential file back into a

program file, and it's a bit more complicated. It isn't very difficult to get the lines of the program on the screen, but in order to tokenize them, you have to press the return key after each line; and you must do this outside of the program.

The following program was written to make this chore automatic. By utilizing a technique called dynamic keyboard action, the computer can terminate a program, perform a task in Immediate mode and then reenter the program—all automatically. It is as if you are programming the computer to program itself.

For practical purposes, the program is very tightly packed, so it's difficult to read. However, for those interested in the techniques involved, which also have many other applications, I've included a detailed explanation of the code.

When you run the Tokenizer program, it automatically reads into memory the sequential file version of your program, tokenizes each line and, finally, erases itself. You are left with only your program stored in the computer, ready for saving and running in the usual manner.

SOME CAUTIONS

Take careful note of the following warnings when you're working with the Tokenizer program:

Tokenizer will not work with programs that are more than 19K or 76 blocks long, because it splits Basic RAM into two parts, one-half of which stores the tokenized program, and the other half the non-tokenized version.

The program that you're tokenizing cannot have any of the same line numbers as the Tokenizer program. If it does, make sure you change them before creating the sequential file.

Here are instructions on converting a sequential file to a program file.

1. Type: LOAD "TOKENIZER", 8 and run it.

2. You will be prompted for the filename of the sequential file you are converting to a program file.

3. After you give it a filename, Tokenizer will read the sequential file into memory. This may take some time, so be patient. Then the program lines will appear, one by one, on the screen as the tokenization is performed. When all the lines have been tokenized, your converted file will be listed. You can then save it or run it. R

Quick Merge!

By Richard De A'Morelli

RUN It Right

C-64; C-128 (in C-64 mode)
Disk drive

If you write your own Basic programs or utilities, you've probably developed several useful subroutines that you frequently want to include in new programs.

For instance, you may have a subroutine that checks the current cursor position to prevent scrolling; another that produces a beep and requests user input; and perhaps another that sets up and Pokes values to your computer's sound chip. The possibilities are endless.

If you write your own programs, you know that the process of typing in Basic lines, checking for typos and debugging is time-consuming and tedious. And having to retype those same lines, again and again, whenever you want to include them in a new program, is even more tiresome.

Wouldn't it be nice if you never had to type, proofread and debug your favorite subroutines more than once; if you had a master program containing dozens of program modules that you could access in a few seconds with just a few keystrokes?

Well, Quick Merge will give it to you!

With this easy-to-use utility, you type in your favorite subroutines only once and save them to disk. You can then, within seconds, retrieve or merge them into any program you're writing.

HOW TO USE QUICK MERGE

Quick Merge works by saving blocks of Basic program lines in the form of sequential files. Through a process that utilizes the C-64's dynamic keyboard feature, you can then easily recall those files and merge them into whatever program you're writing. You don't have to contend with any Peeks and Pokes or remember any memory locations.

Quick Merge is completely menu-driven. Load it by entering

First, load and run Quick Merge. Next, select menu option 10 (Write New Program). Quick Merge will terminate, and brief instructions will appear on the screen.

In Direct mode, type whatever program lines you wish to enter, making sure your line numbers remain below 63900.

For example:

```
10 "{SHFT CLR}HELLO!"
20 PRINT"THIS CREATES A DEMO FILE"
30 PRINT"WHAT IS YOUR NAME?":INPUT N$
40 PRINT"THANK YOU,";N$
50 END
```

Still in Direct mode, type RUN 63900 to reactivate Quick Merge. Choose menu option 2 (Save File), and you'll be prompted to enter a filename. Type "DEMOFILE" as the filename.

In this example, when prompted, type 10 as the starting line number and 50 as the ending line number.

A Quick Merge program file will now be created and stored on disk. After a few moments, the menu will reappear. Select menu option 7 (Directory), and you'll see that a new file called "/DEMOFILE" has been created.

Once a Quick Merge file has been created, you may use any of the other menu commands to view, edit, merge, copy to another disk, print, etc.

Next, select menu option 11 (Exit to Basic) and delete lines 10-50. Type RUN 63900 to restart Quick Merge.

Select menu option 3 (Merge).

Type "DEMOFILE" as the file to be merged.

The file will be displayed on the screen as it is read into memory. When the last character has been read, the number of bytes in the file will be displayed.

Each program line in the file will now flash rapidly on the screen as it is merged into the Quick Merge program. When the final line has been merged, the menu will reappear.

Finally, select menu option 11 again and list lines 10-50. You'll see that they are merged into Quick Merge. You could just as easily have merged a hundred lines or a dozen different program modules!

Table 1. Example of how to use Quick Merge.

```
LOAD"QUICK MERGE",8
```

then type RUN. A menu will appear offering the variety of options described below.

PROGRAM OPTIONS

The first option, View File, lets you examine any existing Quick Merge file. (Of course, the file you select must be on the disk currently in the drive, or a File Not Found error will result.) Text will appear on your screen with line numbers and Basic commands, just as if you were viewing an actual program listing. When the end-of-file marker is reached, you'll be returned to the main menu.

Please note that Quick Merge filenames are always preceded by a slash (/). For instance, a subroutine that's named "Tones" would actually be saved as "/Tones". (You needn't enter the slash; the computer adds it automatically.) This makes it easy to distinguish these files from others on a disk.

The second option, Save File, lets you save any selected block of program lines currently in memory. Before using this option, you must type in some of your own program lines using option #10 (Write New Program).

You'll then be prompted for a unique filename (up to 15 letters). Always try to assign names that are descriptive of a file's content. For example, a subroutine that sets up the SID chip might be called "/Sound"; one that converts numeric strings to true dollar amounts could be called "/Dollar", and so on.

You will then be prompted for the starting and ending line numbers of the block to be saved. The computer checks to ensure that the filename you've chosen does not already exist on your disk. If everything is all right, the program will save, in a sequential file, the program lines you've indicated. When the process is finished, the menu will reappear. You can verify that the file has transferred correctly by using the View option and entering the name of the file just created.

When using Save, you are limited to a maximum file length of 4096 characters. There is no limit to the number of actual program lines that can be saved.

The third option, Merge, is not merely an Append utility; rather, it achieves true merging of program modules into whatever master program is currently in memory. You'll be asked to enter the name of the file to be merged, and the rest is automatic.

The computer ensures that the specified file exists. If it does, it is displayed on the screen while being read into memory. Actually, the file does not consume any Basic memory: It is Poked as ASCII characters into the 4K bank

of free RAM at address range 49152–53247. As each program line appears on the screen, it is automatically merged into your existing program.

This Merge feature is a wonderful, time-saving convenience; but bear in mind that program lines loaded from the new file will overwrite any existing lines having the same number in the current program. Therefore, I suggest that you first view the file to make sure line numbers in that block do not duplicate existing lines. If necessary, change line numbers in your main program so they won't be overwritten.

When the merge is complete, you can exit to Basic and list your program; you will see that the new lines have actually been mixed into the original program! In this manner, you can merge as many of your favorite subroutines as you wish and, within minutes, build an elaborate program containing hundreds or even thousands of bytes, without ever typing a single Basic line or having to proofread or debug a single command!

Once you've begun assembling a library of program files, the Rename menu option permits you to update filenames. You'll be prompted for the old and new filenames, and the file will be renamed accordingly.

When you no longer want a program file, use the Erase option and type the name of the program to be erased. It will be permanently deleted from disk.

If you rewrite an existing file and want to save it in place of the old version, the Rewrite option will do it automatically.

The Directory option will display any disk directory without disturbing the current program in memory. This is helpful when you are searching for a file or when you can't remember the filename you assigned to a program module.

The Duplicate option is a handy feature that lets you copy any Quick Merge sequential file from one disk to another. You'll be prompted for the filename to be copied, and it will be read into memory. You will then be told to insert the destination disk. An exact copy of the original file will be duplicated on the second disk.

The Print option is a convenient feature that sends to your printer the contents of any Quick Merge file. The file is also displayed on the screen as it is printed. When the end-of-file marker is reached, the main menu will reappear.

The Write New Program is the option you must use before actually saving a Quick Merge file. It temporarily halts Quick Merge, and you will be instructed to type in your program lines. When finished, check your typing accuracy and then reactivate Quick

Merge with RUN 63900. You can now use the Save option to create a permanent Quick Merge file of the program lines you just entered, and you'll never have to type those lines again!

Quick Merge has high line numbers, (63900–63999), so as not to interfere with the assembly of your main program. When writing or merging program modules, just be sure to keep your Basic line numbers below 63900. R

A Collector's Cache

By Benjamin Bayman and Philip Bayman

RUN It Right

C-64; C-128 (in C-64 mode)

Most of us have collections of one kind or another—family photographs, stamps, magazine articles, and so forth. If the collection is a small one, it's easy to keep track of every item in it. But a large collection, containing thousands of items, can be difficult to handle.

In order to get maximum pleasure from any collection, you need to have a quick way to locate any particular item in it, or to locate all the items of a particular kind. Find It is designed to help you do these things easily and quickly.

HOW IT WORKS

In this article, we will use the example of a family photograph collection to explain how the Find It program works. Your first step in using this program is to divide your subject into a maximum of 16 categories.

For your family photographs, you might use categories like Mom, Dad, pets, friends, hometown, vacations, home state, and so on. Note that the categories do not have to be mutually exclusive. A given photograph might be assigned to only one category or to any combination of all 16. Choose the categories that fit your particular situation.

Once you have chosen your categories, take out your photographs and number them, starting with 1 and going up to a maximum of 13,568. These numbers are labels by means of which each photograph will be identified. Write each number adjacent to the place where its corresponding photograph is stored.

Now look at each photograph and decide which categories are relevant to it. Write this list of category numbers at some other place near the photograph, say, on its back, if the photographs are not permanently mounted.

Now you have to put all this information into the computer. Soon after you run the program,

the screen will clear and the following menu will appear.

MENU

1. FIND ITEM NUMBERS (SCREEN OUTPUT).
2. FIND ITEM NUMBERS (PRINTED OUTPUT).
3. ADD ITEMS TO THE LIST.
4. CHECK OR CHANGE ITEM CATEGORIES.
5. UPDATE THE DISK FILE.
6. CREATE A NEW DISK FILE.
7. END THIS JOB.

(TYPE 1-7 AND RETURN).

You want to create a new file, so you type 6 and press the return key. You will then be asked for the name you want to attach to the file, the number of categories it has and the name of each category.

With a 40-column screen, a category name can have up to 16 characters. If you exceed your particular limit, those extra characters are ignored. Once you supply this information, a file is created and you are returned to the menu.

Now you must put the data about your collection into the file. When you type 3 from the menu, the screen will clear, and there will appear the message "NEXT ITEM NUMBER IS 1," followed by the list of categories and the pattern shown in Table 1, with the cursor flashing in the lower left-hand corner. Type an X under the category numbers that apply to the first photograph. (In Table 1, categories 1, 5, 14 and 15 have been chosen.)

When you hit the return key, an entry will be made into the list, specifying that item 1 belongs in these categories. The screen will then clear, the message "NEXT ITEM NUMBER IS 2" will appear, followed, as before, by the list of categories and the pattern displayed in Table 1 (without the Xs, of course). Then proceed in the same way as before.

You will find that it takes only a few seconds to enter each item. You need only use the space bar and backspace to move the cursor horizontally, the X key (or any other key) to mark the chosen categories, and the return key.

After you have entered all your items in this way, or have done as many as you feel like doing in this session, hit the return key without putting any Xs or other symbols below the pattern, and the menu will reappear on the screen.

At this point, the safest thing to do is to store on disk the information you have just put into the computer memory. Just type 5 and hit the return key. Your disk drive will start churning. After a minute or two, the menu will reappear, and the information will have been permanently stored.

When you want to use this datafile at some future time, it will be read into the computer from the disk or tape. Note that the Find It program does not itself make any reference to a particular datafile. On a single disk or tape, you could have a copy of this program and several datafiles, each created as described above, storing quite different kinds of information.

WHERE'S THAT PHOTOGRAPH?

Now suppose you want to hunt for a photograph. With the menu on the screen before you, type 1 or 2, depending on whether you want to get the information from the screen or the printer, and hit the return key. The screen will clear and the message "NUMBER OF ITEMS IS 1224" will appear (if your list has 1224 items), followed by the list of categories and the category numbers in the usual pattern.

Suppose you want all the photographs of Mom and Dad at home. Then you type an X, or any other symbol, below the numbers of the appropriate categories (let's say you select 1, 2 and 10). When you hit the return key, you will be asked, "INCLUSIVE OR EXCLUSIVE (TYPE I OR E)"?

In this case, an exclusive search will find all the photographs that have been assigned *only* to categories 1, 2 and 10. You would use this kind of search if you were looking for a particular item that you remember well enough to be certain of all the categories assigned to it.

If you choose an inclusive search, you will be given the numbers of all photographs that have been assigned to categories 1, 2 and 10, regardless of whether these photographs have also been assigned to other categories. For example, photographs of the entire family at home would be listed in an inclusive search with categories 1, 2 and 10, but not with an exclusive search. You would use an inclusive search if you want all the information on a particular subject and don't care what other kinds of information are supplied as well.

Having hit I or E and the return key again, you will be asked to specify "RANGE OF SEARCH." If you want to search the entire list, just press A (for "all"). If, for some reason, you only want to search from items 28 through 635, type 28, 635 and press the return key.

In either case, there will then appear on the screen (or at the printer, if that is what you requested) the identification numbers of the photographs in your list, in the number range you specified, corresponding inclu-

									1	1	1	1	1	1	1
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
X				X									X	X	

Table 1. The pattern of numbers that enables you to specify a set of categories. In this example, the item is in categories 1, 5, 14 and 15.

sively or exclusively to the categories you selected.

If there are no items in the list that satisfy all these criteria, you will be given the message "NO SUCH ITEM". The screen (or printer) will remain constant until you hit the shift key, at which time you will be able to search for another item. After you have done all the searching you want to do, hit the return key without choosing any categories, and the menu will reappear on the screen.

The Find It program gives you a list of the items in your collection that meet the criteria you have specified. To see which ones you really want, you must then go to the actual collection and inspect these items. To speed up this last step, you should be restrictive when you set your criteria; keep the list of items to be inspected as short as possible.

You get maximum efficiency from this scheme if the item numbers themselves carry some information. For example, suppose you numbered your photographs sequentially and with respect to date, with 1 signifying the earliest photograph, 2 the next one, and so on. Then, if you knew the approximate date of the particular photograph you were seeking, you could make a rough guess of what its identification number should be and thereby hasten your search.

MAKING CHANGES

If you suspect that some of the items in the list have been incorrectly categorized, then call up the menu and type 4. After you have pressed the return key, the screen will clear, and the cat-

egory list and its pattern will appear, followed by the words "ITEM NUMBER IS?"

If, for example, you then type 235 and press the return key, X will appear under the numbers in the pattern corresponding to the categories that have been assigned to item 235. There will also appear the message "INSPECT AND/OR ALTER THEN RETURN". If you are satisfied with all the criteria already given to item 235, just hit the return key and you will get a chance to look at the categories of another item.

However, if you want to change the categories of item 235, just move your cursor along and make the desired changes in the locations of the Xs. Check your alterations to make sure they're correct and press the return key. The list in your computer memory will now contain the corrected categories for item 235.

To change the categories assigned to item 235 in your disk's permanent file, type 5 from the menu and press the return key. When you have done all the viewing and altering you want to do, answer 0 when you are asked, "ITEM NUMBER IS?", and the menu will reappear.

It is always a good idea to make a second copy of your permanent file. To do this, just put another disk into your machine, type 5 from the menu and press the return key. R

C-128 Ultra Hi-Res

By David Darus, Ken French and Louis Wallace

RUN It Right

C-128 (in C-128 mode)

In *RUN*'s February 1986 issue, we announced that the C-128 was fully capable of using its 80-column RGB mode for ultra high-resolution (640 x 200 pixels) graphics. We also introduced you to the C-128 Ultra Hi-Res graphics language. In this article, we will expand on that topic with several new additions to the command set.

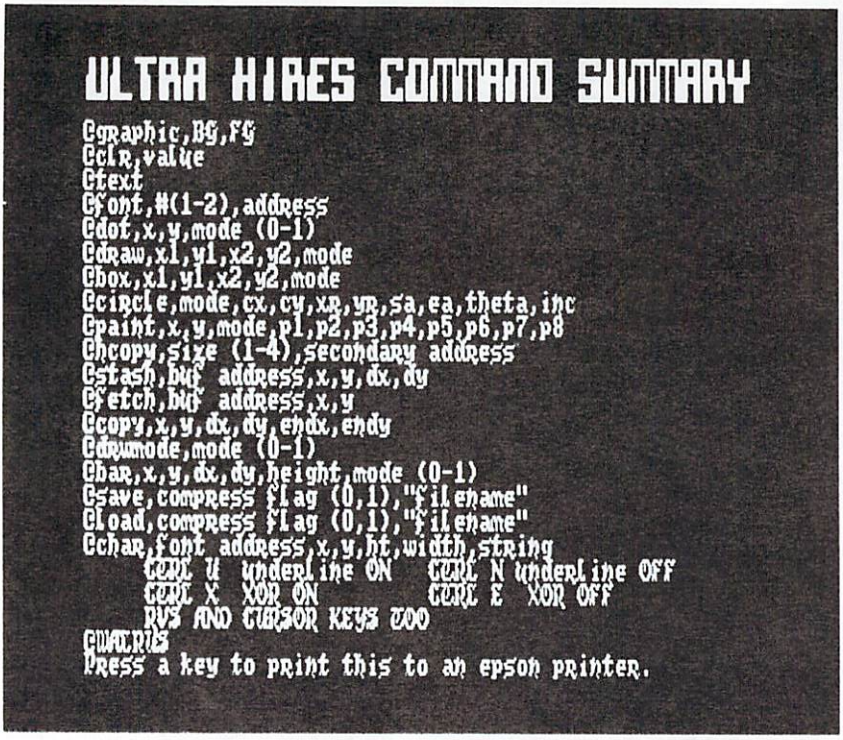
The C-128 Ultra Hi-Res is a language that wedges itself into memory so that its commands are executed along with those of the normal Basic 7.0. Without slowing down normal Basic, it accomplishes this by intercepting the Syntax Error subroutine that issues error messages when you make a typing mistake.

All Ultra Hi-Res commands are preceded by the @ symbol, which is not used by Basic. When the C-128 Basic interpreter encounters this in the context of a command, it passes control to the Syntax Error subroutine, which would normally display a syntax error message and stop the program.

However, we have changed the vector that points to that subroutine. Instead, it points to a machine language subroutine that checks to see if the error is truly an error, or one of our new commands. If the latter, control is passed to the appropriate Ultra Hi-Res module; if not, it jumps to the normal Syntax Error subroutine.

The advantage of this wedge over others is that it allows Basic to run at top speed and checks for our routines only when they are encountered. If you combine this with the Fast command, which operates at 2 MHz, you'll have your C-128 running at full throttle with many new and powerful graphics commands.

To use Ultra Hi-Res, part 2, you'll need a copy of part 1. (See "Ultra Hi-Res Graphics," *RUN*, February 1986, or January/February ReRUN.) You will then



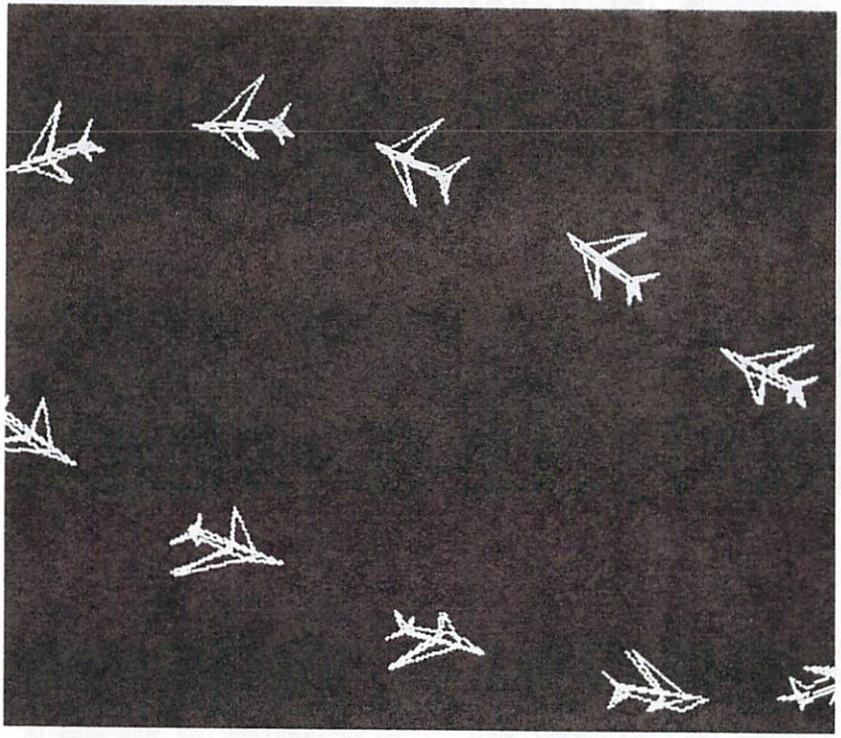
combine all the commands to form an enhanced version (Ultra Hi-Res Version 1.1).

THE ULTRA HI-RES 1.1 PROGRAMS

Three programs accompany this article. Before running them, save them to a disk containing both the old machine language version of the Ultra Hi-Res program and its Boot program.

First, load the program called UH.DATA and run it in normal C-128 80-column mode, not in Ultra Hi-Res mode. It will create a number of binary files on the disk.

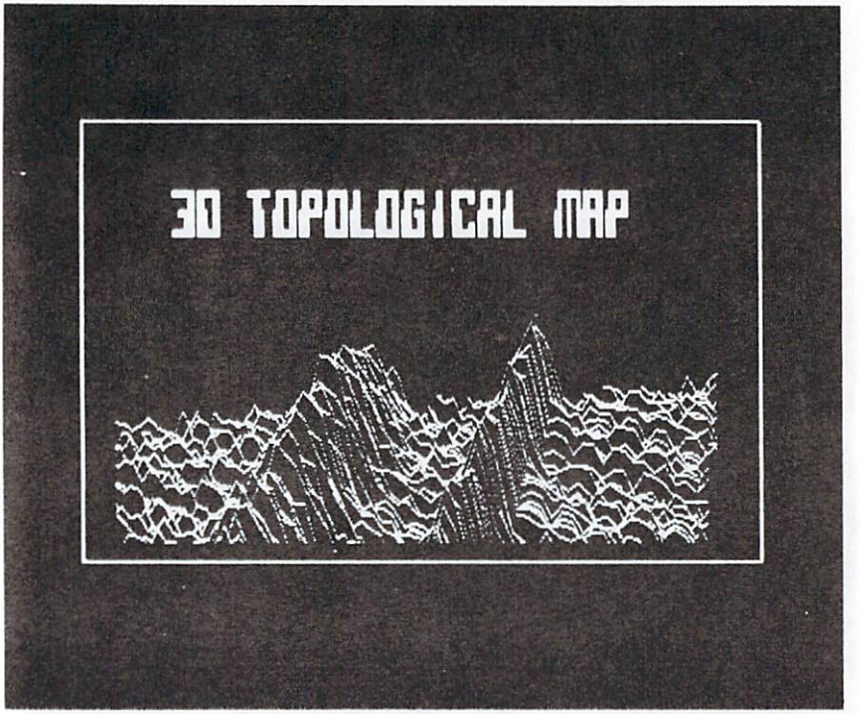
After running UH.DATA, load and run the program called MAKEUH V1.1. This program will first load into memory the old version of Ultra Hi-Res, then it will add the new modules. It will also rename the original version to Ultra Hi-Res.old and save a new version called Ultra Hi-Res. Ultra Hi-Res 1.1 is now ready for use. To activate it, load and run



the Ultra Hi-Res Boot program, as you used it to load the original Ultra Hi-Res program.

The final program is called UH.PIC CONVERT. We have changed the original picture-file format to allow upward compatibility of Ultra Hi-Res pictures in future applications programs. UH.PIC CONVERT allows you to change pictures made with version 1.0 to this new format. Insert a disk that contains the pictures you want converted and answer the prompts. Your old Ultra Hi-Res pictures will now be fully compatible with Ultra Hi-Res 1.1.

Finally, to legally use this program to create applications you wish to give away, you must use the fifth new command, @WALRUS, which creates a logo crediting the program's authors. It is your legal obligation to display this logo for any non-personal use. (If you intend to sell your applications, please contact the authors about licensing.)



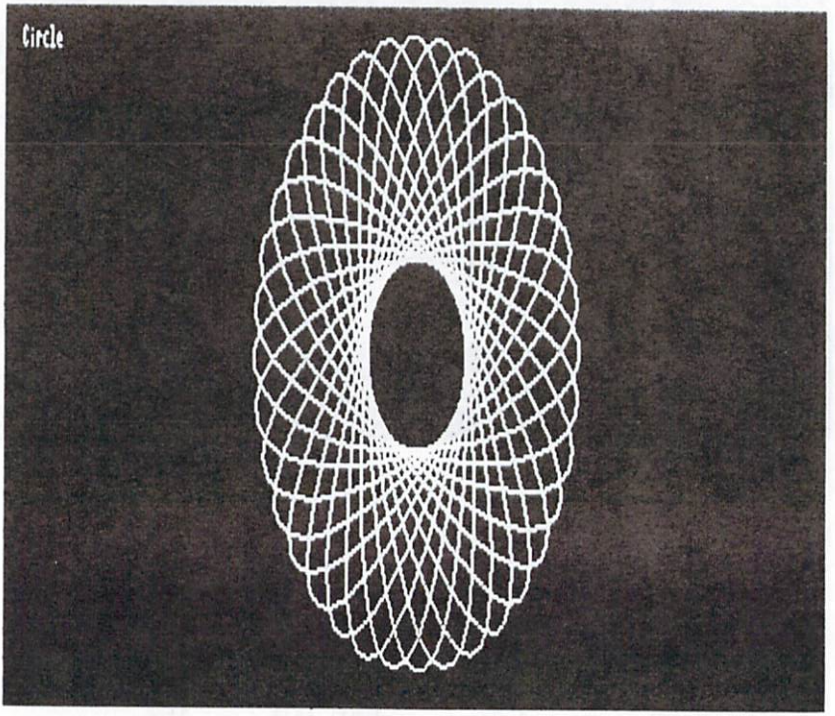
THE COMMANDS

Part 2 adds five new commands that give professional-level graphics power to the already-powerful Ultra Hi-Res command set. Combined with the C-128's large memory and 2-MHz clock speed, you will be able to use Basic to write very impressive applications programs.

The first new command—@CIRCLE—is also found in normal 40-column Composite mode and functions in the same way; however, with Ultra Hi-Res 1.1, you can only have two colors on screen at once, rather than the 16 available in 40-column mode. To compensate for this loss of color, you can change your drawing mode from Draw to Erase. The syntax is:

@CIRCLE,mode,cx,cy,xr,yr<,
sa,ea,angle,increment>

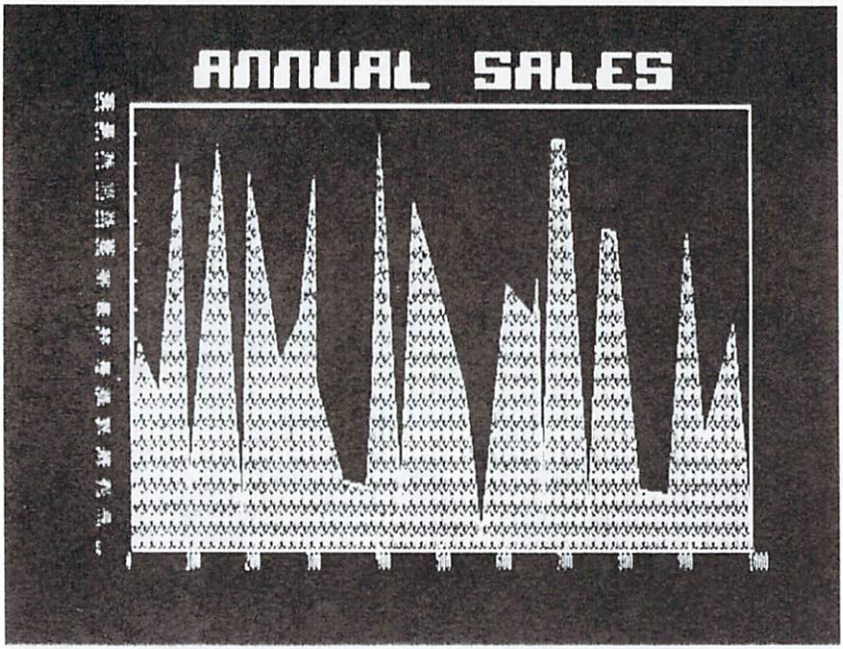
Mode is 0 for erase and 1 for draw; cx is the x coordinate of



the center (0–639); cy is the y coordinate of the center (0–199); xr is the x radius (1–319); yr is the y radius (1–100). The parameters within the brackets, < >, are optional. The sa is the starting angle (0–360); ea is the ending angle (0–360). These allow you to draw arcs easily. The angle is the number of degrees (0–360) you wish to rotate the circle, ellipse or arc. The increment is the number of degrees used in drawing the circle. By changing the increment, you can use the Circle command to draw polygons.

The defaults for the optional parameters are: sa = 0; ea = 360; angle = 0; and increment = 8. For more information on the circle parameters, see the C-128's system guide.

The second new command is called @PAINT. Unlike the 40-column Paint command, this one only provides one color at a time. This could be a problem when it comes to graphics and charts.



The @Paint command compensates for this paucity of colors by allowing you to fill areas with varieties of different patterns. The syntax of the command is:

@PAINT,x,y,mode<,p1>
<,p2><,p3,p4>
<,p5,p6,p7,p8>

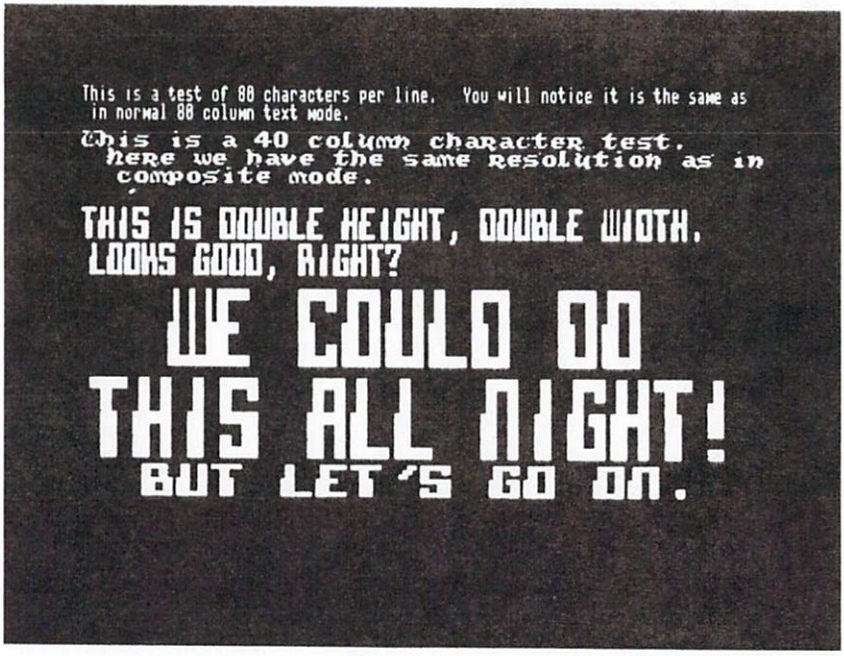
Here x is any value from 0–639; y is any value from 0–199; and mode is 0 (for erase) and 1 (for draw). The p values are numbers from 0–255 and represent the decimal values of the binary patterns used in the fill. For example, if you wanted to generate an interesting patchwork pattern to fill an area, you would use the following values.

01100110	102
11001100	204
00110011	51
10000001	129
10011001	153
01111110	126
00000000	0
10101010	170

The command format is:

@PAINT,x,y,1,102,204,51,129,
153,126,0,170

This will fill a specified area with that pattern. If you do not



enter any specified values, the area will be filled with a solid pattern. By entering only one number, its pattern is repeated eight times; if you enter two numbers, the pattern of the pair is repeated four times; if you enter four numbers, each pattern is repeated once.

You can clear a solid area (not a pattern) by using a mode of 0. You must make sure the area to be filled is completely enclosed or the pattern will leak out and fill the screen.

The third command is @HCOPY. This will dump the graphics screen to a printer in any of four different sizes. The syntax is:

@HCOPY,size, secondary address

The size is from 1-4, and the secondary address is whatever your printer interface requires for Graphics mode with no linefeeds. For example, the Cardco B and PPI with Graphics interfaces require a secondary address of 5. At the moment, only the Epson, Mannesmann Tally Spirit 80 and Olivetti PR2300 printers are supported, but ad-

ditional modules will be added if the demand is there.

The fourth command is @DRWMOD. This sets up a special form of the Draw mode that performs an XOR on the screen when you use any of the drawing commands (Dot, Line, Box, Circle or Bar). It reverses lines on the screen. This command allows you to create the illusion of transparency when two points overlap. The syntax is:

@DRWMOD,mode

The mode is 0 or 1, with 0 indicating no complement and 1 indicating complement.

A GEM OF A PROGRAM

You don't often find a graphics utility that enhances the usefulness of a computer for personal and professional applications. Ultra Hi-Res is a rarity that greatly extends the already-powerful features of the C-128. You will most likely discover many ways to use this new graphics power. R

Give Your C-128 More Character

By Frederick Goddard

RUN It Right

C-128 (in 80-column mode)
Disk drive

This program, 80-Column Characters, lets you create your own character set, which can then be used in both 40- and 80-column modes and in the C-64 mode. First, I'll explain how the C-128 operates in 80-column mode. Then I'll provide a short Basic subroutine that allows you to use custom characters with 80 columns. Finally, I'll describe the 80-Column Characters program.

Although the *Commodore 128 System Guide* does not say so, it is actually easier to change the character set for the 80-column than for the 40-column mode. In the 40-column mode, the C-128 uses the same VIC chip as the C-64. This chip operates through the interrupt routine and reads the character set from memory 60 times each second.

This is a disadvantage when you try to design and use customized character sets in 40-column mode. You cannot run your program in Fast mode; you can only see 256 characters on the screen at any one time (128 characters each in normal and reverse); and you must normally give up 2K or more of memory to provide RAM storage for the character set as long as you use it.

80-COLUMN SCREEN MODE

In 80-column mode, the C-128 operates in a completely different manner and without any of the drawbacks mentioned above. The 80-column mode is controlled by a separate processor, the 8563, which has its own independent memory. This processor displays all 512 characters on the screen at the same time. In addition, it provides for optional underlining of any character and automatically generates either a

block (reverse) or underline cursor. This means you don't have to design reverse or underline characters in your set of 512; the 8563 does it for you.

Since you can have 512 different characters on the screen at the same time, and a normal full ASCII upper- and lowercase text set is only 96 characters, it is possible to design five different character fonts and display them on the screen simultaneously. For example, you could have regular, boldface, italic, italic-boldface and superscript character sets, all with optional underlining.

Since the 8563 chip has its own memory, you need to allocate RAM for your character sets only temporarily. On startup, the C-128's 6510 microprocessor sends the character set data (the same ROM characters used by the VIC chip) to the 8563 through an initialization routine. The 8563 stores this information in its own memory. You can, through this initialization routine, load your own custom character set into the 8563 memory, replacing the original set and freeing for other uses the RAM memory you had been using.

CHANGING 80-COLUMN CHARACTERS

The initialization subroutine is used in the 80-Column Character program and is found in lines 40–80. This short, five-line Basic subroutine loads the character set from disk and creates and runs a machine language subroutine to perform the 8563 initialization.

The C-128 memory-management system uses 16 address banks to keep track of the various areas of ROM and RAM. This bank address system is described on page 370 of the C-128's system guide.

For the present discussion, it's important for you to know that bank 0 addresses the 64K of RAM that holds operating-system variables, 40-column screen memory and your Basic program text; bank 1 addresses 60K of RAM reserved for variables and strings; bank 14 addresses character ROM; and bank 15 addresses the Kernal routines that control the C-128 operating system. The machine language subroutine uses the existing Kernal routine found at $FF62 (65378 decimal) in ROM bank 15.

I could write a long machine language program that duplicates this ROM routine. Instead, I've written a very short wedge that uses the ROM routine but fools it into getting its data from RAM bank 1 instead of ROM bank 14. The Basic subroutine to load and initialize a custom character set is as follows.

```
20 POKE58,DEC("D0"):CLR:GOSUB30:
   GOTO90
30 BLOAD"CHRSET80",B1,P53248
40 RESTORE60:FORI = 0TODEC("2B"):
   READX$:POKEDEC("0200") + I,
   DEC(X$):NEXT
50 BANK15:SYSDEC("0200"):RETURN
60 DATA A9,20,8D,A2,02,A9,22,8D,
   A3,02,A9,02,8D,A4,02,20,62,FF
70 DATA A9,AD,8D,A2,02,A9,00,
   8D,A3,02,A9,FF,8D,A4,02,60
80 DATA A2,01,BD,F0,F7,AA,AD,
   00,FF,60
90 POKE58,DEC("FF"):CLR
```

You should place this Basic subroutine at the beginning of any Basic program that uses 80-column custom characters. Line 20 changes the top-of-memory in RAM bank 1 so that you can safely use memory starting at $D000 (decimal 53248) to make temporary space for the data. (Line 90 restores the memory pointer, giving you back all of bank 1 as free memory.) Line 30 loads the custom character set; line 40 reads the short machine language program from the Data statements and puts it at starting address $0200; and line 50 runs the machine language program.

As shown above, you should run the subroutine first in your program, as any Basic variables will be lost by the action of lines 20 and 90. If you want to run this initialization routine later in your program, you should set the top of memory (line 20) at the beginning of your program and omit line 90. This means that you will lose 11K of variable storage in bank 1. To restore the original ROM character set, use SYS65378.

DESIGNING CUSTOM CHARACTERS

When you run 80-Column Characters, the Help screen appears on the bottom half of the display. It contains all the instructions for using the program, and you may recall it at any time by pressing the help key. The function keys are redefined so that f1 loads a character set file from disk and f3 copies the ROM characters to disk. Since you will have no character set on disk at the beginning, press f3 and create one. Then use f1 to load that character set file.

After you've loaded the character set into RAM, all 512 characters will be printed in eight rows of 64 characters each at the top of the screen. Every two rows of characters represent a set of 128 characters. The first two sets (rows 1–4) are the characters obtained by the command Print CHR$(142); rows 3–4 are printed when you are in Reverse mode (CTRL 9). The last two sets (rows 5–8) are the characters obtained by the command Print CHR$(14).

A blinking cursor appears over the @ character at the top left-

hand corner of the screen. To redesign a character, position the cursor over that character and press the return key.

As an alternative to using the cursor keys, you can simply press the key you wish to redesign. Select uppercase or graphics characters by pressing the shift or Commodore key as you make your key selection.

After you select the character you wish to redesign, it will appear in an 8 × 8 yellow box at the top right-hand corner of the screen. Each cell in the box represents one pixel (dot) in the screen character. Yellow cells represent pixel-off, and brown cells represent pixel-on.

You can move the cursor in the character-design box with the cursor keys or the 8, 4, 6 and 2 keys on the numeric keypad. Pressing the plus sign or period keys turns a pixel on, and pressing the minus sign or zero key or the space bar turns a pixel off. Pressing R causes the character design to be reversed. When you have redesigned the character, press the return or enter key to initialize it; then select another character or a command.

You can test the appearance of your redesigned characters in a Test Type mode by pressing f7. This allows you to type into the window at the bottom of the screen, and almost all of the usual keyboard commands will work. For example, you can change the character sets by pressing shift-Commodore; print reverse character sets by pressing CTRL-9; change colors; activate underlining by pressing CTRL-B; clear the screen window; and so on. You exit from this mode by pressing f7.

When you are finished redesigning a character set, press f5 to save your new characters to a disk file. You can later use these characters in the programs you write by using the BLoad command and the initialization subroutine in lines 40–80 to load the character set file into RAM. [R]

Murder by Byte

By Christopher Rose

RUN It Right

C-128 mode (80-column)
Joystick; printer

A murder has been committed, and it's your job to solve it. To begin your investigation, you—and any other players acting as competing detectives—step into an old English manor from which a body has just been removed. You must search the house for weapons and question the ten suspects. But only one of them is a killer, and some of them will lie to you. Who is the murderer, and what weapon did he or she use? These are the questions you must answer in the case of Murder by Byte.

To play this game, you'll need a Commodore 128, a monitor that will display the 80-column mode and a joystick to input your moves. You will also need a printer for printing out gamepads. The program uses only standard ASCII characters, so any printer that prints 80 columns across should do.

How To Play

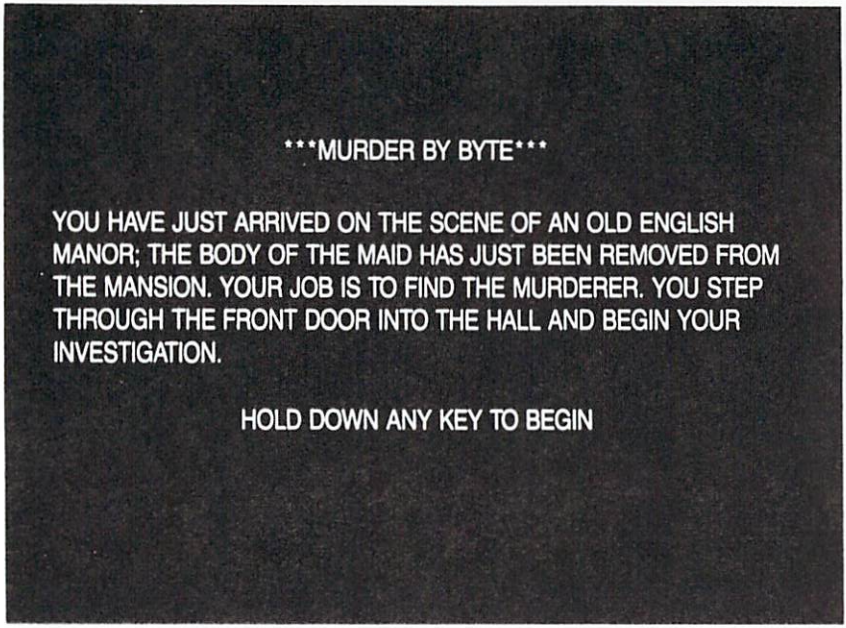
Load "Murder Mystery" and type RUN. The computer will first ask for players' names (a maximum of five). Type in the name of each player, followed by a return. When you're through, press the return key a final time. The screen will clear, and the opening display will appear. After reading the display, press any key; after a few moments the game will start.

From here on, the game is played entirely with the joystick (plugged into port number 2). Move the stick up and down to move the pointer, and press the button to make choices.

The playing board is composed of the fourteen rooms of the house. In these rooms, you'll look for missing weapons and question the suspects about one another.

The game is played through a series of menus. The main menu shows the moves you and any other players may make. Here are the options:

A. Question a suspect who is



in the room. When you select this option, another menu will appear, asking whom you wish to question the suspect about. The suspect will then say what he or she thinks about that person.

Most of the strategy lies in questioning the suspects. From what they say, you'll be able to find the murderer. Here is a step-by-step explanation of this process.

There are ten suspects in the house, each in an separate room. A suspect may leave a room to enter an unoccupied room. You can question any suspect who happens to be in the same room with you.

Before beginning to understand how to find the murderer, you have to know a few things about the suspects. There are two types of people in the house: truthtellers and liars. A truthteller will attempt to give honest opinions about other suspects, whereas a liar will always lie about the others. However, whether or not a person is a liar has *no* bearing on whether he or she is a murderer.

Truthtellers do not like the murderer, but they also may not like several other people. In any case, the murderer is among those a truthteller dislikes. With a

liar, the reverse is true: anyone a liar likes may be the murderer. Of course, you must also consider the person you are questioning as a possible murderer. By the way, each suspect will always say something nice about *himself*, even, or perhaps especially, the murderer.

To distinguish between truthtellers and liars, you must carefully examine their responses. If a person says mostly good things about the other suspects, he is probably a truthteller. If he says mostly bad things, he's probably a liar.

Once you've determined a suspect's status, you may be able to narrow down the field by seeing what he says about others, as described above. But beware; some suspects will say the same number of good things as bad things, thus rendering their responses useless.

In rare instances, a person will appear to be of one type, whereas he's actually the other, and this can lead you to the wrong conclusion. Obviously, the more people you ask a suspect about, the clearer his status will become.

B. Examine the furniture in the room. Select this option to find the murder weapon, one of five weapons available. The actual murder weapon has been removed from the house; the other four are hidden in the furniture, so search the house. Once you find them, the missing fifth must be the murder weapon.

C. Move to a connecting room. This option displays another menu of possible rooms to select. You can move into any room that's connected by a door to the one you're already in.

D. Solve the murder. This selection displays two menus, from which you choose your suspected murderer and murder weapon. If you're correct, the game will congratulate you and end. If you've guessed wrong, the program will tell you so. If there are other players, the game will then continue with the next in turn; but if you're playing alone, the game will end after telling you that you lost.

E. View a map of the house. You may choose this option without forfeiting your turn. It displays a map showing the rooms and the doors between them. Use it to decide where to move.

Since there's so much information you must keep in mind during the game, I've supplied a second program, Mystery Gamepad. The gamepad this program prints out will help you keep track of the people you've questioned and those you have asked them about. It will also help you recall where you've searched for weapons.

Each game is set up randomly at the beginning, making each

one different. The victim, murderer, murder weapon, suspects' responses and weapon locations will all change from game to game. This random selection may make one game fairly straightforward and the next very difficult. You never know until you begin.

Play it a few times to get the hang of it, referring to the instructions as you go. If you get stumped and can't solve the mystery, break out of the program and type

```
? S$(M),W$(MW) <return>
```

This will tell you who did it and with what. After a few games, you should be ready to do some serious sleuthing. R

Please send me back issues of ReRUN

____ Spring Edition
____ Gamepak
____ Summer Edition
____ Fall Edition
____ Productivity Pak (Disk only)
____ Winter Edition
____ Jan./Feb. 1986 (Disk only)
____ March/April 1986 (Disk only)

____ C-64
____ VIC-20

____ Cassette version(s) at $11.47*
____ Disk version(s) at $21.47

** Prices include postage and handling. Foreign Air Mail please add U.S. $1.50 per item and $25 per subscription. Prepayment only.*

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Name

Address

City State Zip

Signature

ReRUN • Elm Street • Peterborough, NH • 03458

BEAT THE RUSH!

Please send me:

☐ **1 year (6 issues) for $89.97**
☐ **July/August ReRUN disk for $21.47.***

***Available in August.**

Programs run on C-64 and C-128 (in C-64 mode).

Price includes postage & handling. Foreign Air Mail please add U.S. $1.50 per item and $25 per subscription. Prepayment only.

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Name

Address

City State Zip

Signature

ReRUN • Elm Street • Peterborough, NH • 03458

11 RUN *Programs Included on this Disk:*

Graphics ▸ Disk Utilities ▸ Games ▸ Education ▸ Applications ▸ Programming Aids

From the May RUN:

- ▸ Ultra Hi-Res Graphics, Part 2
- ▸ Swish!
- ▸ Disk Reader
- ▸ Label Maker
- ▸ Easy Data Entry

From the June RUN:

- ▸ Arithme-Sketch
- ▸ Quick Merge
- ▸ Easy Disk-File Conversion
- ▸ Give Your C-128 More Character

PLUS Bonus Programs:

- ▸ Murder By Byte
- ▸ Collector's Cache

If any manufacturing defect becomes apparent, the defective disk will be replaced free of charge if returned by prepaid mail within 30 days of purchase. Send it, with a letter specifying the defect, to:

ReRUN • 80 Pine Street • Peterborough, NH 03458

Replacements will not be made if the disk has been altered, repaired or misused through negligence, or if it shows signs of excessive wear or is damaged by equipment.

The programs in ReRUN are taken directly from listings prepared to accompany articles in *RUN* magazine. They will not run under all system configurations. Use the RUN It Right information included with each article as your guide.